Technical Bulletin

Part No. 74-0116

DataStage Sort

This technical bulletin describes Release 1.2 of the DataStage Sort stage. The Sort stage adds high-performance data sorting capabilities to DataStage.

Printing History

First Edition (74-0116) for Release 1.0, March 1998
Second Edition (74-0116) for Release 1.1, December 1998
Third Edition (74-0116) for Release 1.2, March 1999
Updated for Release 1.2, August 2002
Updated for Release 1.2, August 2003

How to Order Technical Documents

To order copies of documents, contact your local Ascential subsidiary or distributor, or call our main office at (508) 366-3888.

Documentation Team: Marie E. Hedin

Introduction

This technical bulletin describes the following topics for Release 1.2 of the Sort stage, updated for DataStage Release 7.0:

- Functionality
- Configuration requirements
- Configurable properties
- Sort criteria
- Stage properties
- Problem fixed

DataStage Sort is an active stage that sorts a variety of data. It sorts small amounts of data efficiently in memory when there is enough main memory available. It sorts large amounts of data using temporary disk storage, not virtual memory swap space.

The model for the Sort stage is the UNIX *sort* command, as used in a shell pipeline. Input data rows to be sorted arrive as lines of ASCII characters read from the *stdin* stream. You use command line arguments to specify how to sort these rows. The resulting sorted rows are written as lines of ASCII characters to the *stdout* stream.

In DataStage, the Sort stage receives a stream of rows using a single input link. The rows are already separated into individual column values. The values for the stage properties and column attributes specify how to sort these rows. The resulting sorted rows are written as column values to a single output link.

The Sort stage must have one input and one output link. Considerations for the columns in the rows for the input and output links include the following:

- A single input stream link provides rows of data to be sorted. The column type of the input column must be convertible to the type of the output column.
- A single output stream link receives sorted rows of data. Output rows have the same column order as input columns. The names of output columns may differ from the names of input columns.

The output link data type for each column determines the type of comparison to perform:

- Numeric comparison for numbers
- Date and time
- Character string (left to right sort for strings and timestamps)

Functionality

The Sort stage has the following functionality and benefits:

- Supports NLS (National Language Support). For more information, see *DataStage NLS Guide*.
- Supports MetaStage. For more information, see *MetaStage User's Guide*.
- Supports an option to sort per column using a collating sequence map.
- Supports an option to request a stable sort. A stable sort preserves the input order of rows that compare as equal.
- Logs messages to report nonfatal warnings that can impact loss of precision of sorted data.
- Supports performance tuning parameters for efficient sorting, thus limiting virtual memory use.

The following functionality is not supported:

- Bulk loading for stream input links
- Stored procedures

Installing the Plug-In

For instructions and information supporting the installation, see *DataStage Plug-In Installation and Configuration Guide*.

Configurable Properties

You can configure the following properties to improve performance for the Sort stage:

- Max Rows in Virtual Memory
- Max Open Files

Max Rows in Virtual Memory Property

The Max Rows in Virtual Memory property lets the job designer regulate the amount of data in virtual memory. By limiting the total number of rows to sort, the sort algorithm performs incremental sorts. This reduces the virtual memory usage and excessive page swapping that occurs when you have a large amount of input data associated with the input link.

This property is used when the number of rows within the input link exceeds the supplied value for this property. The sort algorithm sorts rows in multiples of this value and stores these sorted groups of rows in temporary files. These temporary files are then merged together for the final sort.

Max Open Files Property

The Max Open Files property limits the number of intermediate data files that are created when incremental sorts are performed. The processing of the data is controlled by the following:

- The Max Open Files property
- The Max Rows in Virtual Memory property
- The actual number of rows associated with the input link

Example

Assume that the input link contains 100,000 rows of data, and Max Rows in Virtual Memory is set to 10,000 rows.

The sort algorithm reads in the first 10,000 rows from the input link, performs an intermediate sort, then stores the sorted data to a temporary file. The algorithm continues to group 10,000-row chunks from the input link, storing the sorted results in unique temporary files, until one of the following conditions is met:

- All of the input data has been processed into temporary files. The total number of temporary files is less than the value specified in the Max Open Files property.

  After the intermediate sorts, the 10 temporary files are merged and sorted together, resulting in the final sort that is written to the output link.

- The number of temporary files equals the value specified in Max Open Files.

  If, for example, Max Open Files is set to 5, the first 50,000 rows are processed as five temporary files, with 10,000 rows each. These temporary files are merged together to form a new temporary file with 50,000 rows of sorted data. The algorithm grabs the next 10,000 rows from the input link and continues with the intermediate sorts. This algorithm continues recursively until all the data is processed.

Note: If the values of these parameters are too restrictive, a high number of intermediate sorts results with constant file merging.

Sort Criteria

The Sort stage accumulates input rows in memory, limited by the Max Rows in Virtual Memory property. It sorts the accumulated rows, storing them in disk files, if necessary. (Small sort sets can be sorted in memory.) It merges these stored files and writes the rows to the output link.

You can enter the values listed in the following table to specify the order of rows, depending on case-sensitivity:

Case-Sensitivity	Ascending Order	Descending Order
Sensitive	a asc ascending	d dsc descending
Insensitive	A ASC ASCENDING	D DSC DESCENDING

The following example specifies to sort the resulting rows in case-sensitive ascending order on the input link REGION column. It uses an external map file named CSM in the C:\USER directory on the CUSTOMER column, and descending order on the SALE_PRICE column (see the Sort Specifications property in “Stage Properties” on page 5).

```
REGION asc, CUSTOMER ASC C:\USER\CSM, SALE_PRICE DSC
```

Collating Sequence Maps

You can specify collating sequence map sorting per column. The format of the map accommodates character encoding, such as single-byte, double-byte, and variable number of bytes. You can specify a separate map file for each column to be sorted. The map file is used in sorting character string values in that column. The map does not affect the sorting of noncharacter-string values, that is, numeric, date, time, and timestamp values.

A collating sequence map is a comma-delimited file containing two columns. The left column is a single character code (in a single- or multibyte encoding, as appropriate). Use an escape character to enter delimiter characters and arbitrary byte values.

The right column is an integer value using ASCII characters for the decimal digits. The column contains the numeric weight used when comparing corresponding characters in two strings. The lower the number, the earlier it sorts. If two

characters have identical weights, they compare as equals. Any character not in the map compares higher than any character in the map. For example, the following sequence map contains these comma-delimited columns:

```
a,3
b,3
c,3
d,5
g,6
e,1
```

You can, for example, provide a collating sequence map to specify the collating sequence for the French alphabet.

Stage Properties

The following table includes these column heads:

- **Prompt** is the text that the job designer sees in the stage editor user interface.
- **Default** is the text used if the job designer does not supply any value.
- **Description** describes the properties.

The Sort stage supports the following stage properties:

Prompt	Default	Description
Sort Specifications	None	The criteria by which the ASCII characters in the rows read from the input link are sorted. See "Sort Criteria" on page 4 or more information.
Max Rows in Virtual Memory	10,000	The maximum number of rows (from 2 to 50,000) that can be sorted in virtual memory. The smaller the row, the more rows that can be sorted.
Temporary Directory	None	The pathname where the temporary files that are created during the sort are stored. If you do not specify a pathname, the current working directory in the server is used.
Escape Character	\ (backslash)	The single character used in the collating sequence map files to specify control characters.

Prompt	Default	Description
Tracing Level	0	Controls the type of tracing information that is added to the log. The available tracing levels are: 0 No tracing 1 Stage properties 2 Performance 4 Important events You can combine the tracing levels. For example, a tracing level of 3 means that stage properties and performance messages are added to the log.
Stable Sort	No	Indicates whether the sort is a stable sort. A stable sort preserves the order of the input rows that compare equal.
Column Separator	, (comma)	The single character separating the two columns in each line of the collating sequence map file.
Max Open Files	10	The maximum number of files that can be open simultaneously. The larger the value, the better the performance. When using one or more of these stage instances in the job, the total number of open files of all the stage instances must not exceed 20.